COURSE DESCRIPTION CARD - SYLLABUS

Course name
Fundamentals of Computer Science [S1AiR1>PI]

Course

Field of study
Automatic Control and Robotics

Year/Semester
1/1

Area of study (specialization)
–

Profile of study
general academic

Level of study
first-cycle

Course offered in
polish

Form of study
full-time

Requirements
compulsory

Number of hours

Lecture
30

Laboratory classes
30

Other (e.g. online)
0

Tutorials
0

Projects/seminars
0

Number of credit points

5,00

Coordinators

dr hab. inż. Jakub Kołota
jakub.kolota@put.poznan.pl

Lecturers

mgr inż. Paweł Czopek
pawel.czopek@put.poznan.pl

dr inż. Krzysztof Kolanowski
krzysztof.kolanowski@put.poznan.pl

dr hab. inż. Jakub Kołota
jakub.kolota@put.poznan.pl

dr inż. Janusz Pochmara
janusz.pochmara@put.poznan.pl

dr inż. Adam Turkot
adam.turkot@put.poznan.pl

Prerequisites

The student starting this subject should have a basic knowledge of computer equipment and its operation. Should have the ability to solve basic problems in the area of obtaining information from the indicated sources. He should also understand the need to broaden his competences as well as be ready to cooperate within a team. In addition, in the field of social competence, the student must present attitudes such as honesty, responsibility, perseverance, cognitive curiosity, creativity, personal culture, respect for other people

Course objective

Acquainted with the methodology and principles of structured and object-oriented programming using the C ++ programming language. Developing students' ability to solve problems in the area of modeling and implementation of information systems. Students learn to simulate and analyze the operation of object-oriented IT programs and plan and document the IT work done. Developing programming skills in students. Creating awareness of the need for a professional approach to technical issues, meticulous reading of IT systems documentation. The student learns to set goals and define priorities leading to the task through structural and object-oriented implementation of the code.

Course-related learning outcomes

Knowledge
K1_W8, K1_W9
- has ordered knowledge of selected algorithms and data structures as well as methodologies and techniques of procedural and object-oriented programming
Skills
K1_U1
- is able to construct an algorithm for a simple engineering task and implement, test and run it in a selected programming environment on a PC for selected operating systems
Social competences
K1_K2, K1_K5
- understands the need and knows the possibilities of continuous training, raising professional, personal and social competences

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

Summative rating:
a) in the scope of lectures, verification of assumed learning outcomes is carried out on the basis of answers to questions about the material discussed during the lectures (code implementation verified on a PC in the CodeBlocks programming environment and Microsoft Visual Studio). The exam includes an individual approach to each examined student and analysis of the code written during the exam being the solution to the assigned task. If you get a high grade from the final 4.0-5.0 laboratory, the teacher can prescribe it as an examination part.
b) in the scope of laboratories, verification of assumed learning outcomes is carried out by a practical computer test and the final design of the object-oriented application. The final grade is issued as the average grade obtained from the final project and colloquium. If some of the laboratory classes are not carried out for reasons beyond the control of the lecturer (e.g. rector's hours), the lecturer may resign from conducting the test. Then the final grade is defined on the basis of the project and activity in the classroom.

Programme content

The lecture program includes the following issues:
During the semester, the lecturer comprehensively discusses semantics of the C ++ language in structural and object-oriented terms. Each lecture is dedicated to a different issue and presents solutions in the form of ready implementations. Students receive lecture materials in the form of files containing ready source files along with the code discussed during a given lecture on the e-learning platform. The content of the lecture is supplemented with lectures on Unified Modeling Language (UML) modeling, during which the student becomes familiar with the method of creating documentation and UML notation (class and object diagram, activity diagram, sequence diagram, use case diagram, etc.). Laboratory exercises prepared have been in the form of pdf files, which are tasks that you can do yourself during classes. For this purpose, the student uses the student's account on the Moodle e-learning portal (https://moodle.put.poznan.pl) and downloads the document with the task content. During the class, the teacher briefly recalls the program content related to the given exercise (discussed earlier by the teacher during the lecture). Further laboratories cover the followingprogrammingissues:
C ++ language - introduction to language syntax and basic instructions; class and object - quantifiers of the private and public class sections; constructor (copy constructor), destructor, class static components (static modifier); pointers, dynamic arrays (moving around and dynamically allocating / releasing it); inheritance (multilevel inheritance issues) and mechanism of friend functions; virtual methods and the issue of polymorphism; class abstraction; overloading functions and operators; casting and type conversion; STL,

list container (list) and array container (vector); graphical interface of the window application; implementation and defense of own project
Additional content of the lectures are interesting and inspiring issues proposed by students during the semester, which are then discussed in the form of presentations at the last lecture in the semester.

Teaching methods

lecture: multimedia presentation covering the implementation of object-oriented programming issues discussed in a given lecture (all materials developed electronically and embedded on an e-learning platform)
laboratory exercises: practical exercises, implementation of tasks and algorithms defined in studies entrusted to the student, discussion on the complexity and optimization of the code during classes (all materials developed electronically and embedded on the e-learning platform)

Bibliography

Basic
1. D. Vandevoorde, J. Mincer-Daszkiewicz, Język C++ : ćwiczenia i rozwiązania, Wyd. Naukowo-Techniczne, 2001
2. K. Walczak, Nauka programowania obiektowego w języku C++, Wydaw. W&W, 2004.
3. B. Stroustrup, Język C++, wydanie V, WNT, Warszawa 2000
4. Jerzy Grębosz, Symfonia C ++ Standard, Editions 2000, Kraków 2005
5. P. Paczuski, Programowanie w C++ : szkoła pisania programów od podstaw, Springer Polska, 2005.
6. Jerzy Kisielewicz, Język C++ Programowanie obiektowe, Oficyna Wydawnicza Politechniki Wrocławskiej, 2005
Additional
1. B. Eckel, Thinking In C++, Edycja polska, Wydawnictwo Helion
2. W. Gryglewicz-Kacerka, A. Duraj, Projektowanie obiektowe systemów informatycznych. Wydawnictwo PWSZ Włocławek, 201

Breakdown of average student's workload

	Hours	ECTS
Total workload	125	5,00
Classes requiring direct contact with the teacher	63	2,50
Student's own work (literature studies, preparation for laboratory classes/ tutorials, preparation for tests/exam, project preparation)	62	2,50